Plan de análisis de capacidad

MISW 4204 – Desarrollo de software en la nube
Simón Buriticá
Jhonn Sebastian Calderón Bravo
Diego Andrés Naranjo Ríos
Juan Pablo Rodríguez García

Universidad de los Andes - 2024

Contexto:

La funcionalidad de cargar y editar videos es la de mayor valor para los pilotos que deseen participar en la competencia, al igual que la de mayor complejidad debido a la cantidad de subprocesos y recursos que consume. Por tal razón, poder determinar la capacidad de procesamiento de tareas de edición en simultaneo y las condiciones de operación ideales que permitan obtener la máxima salida de videos (throughput) es crucial para ofrecer a los usuarios la mejor disponibilidad y latencia posible.

Objetivos:

1. Determinar la cantidad óptima de videos a procesar en simultaneo
2. Calcular el tiempo promedio de procesamiento de un video desde su carga en minutos
3. Medir el porcentaje de error de solicitudes para varios escenarios de procesamiento de video

Criterios de aceptación:

Para el contexto de las pruebas, se definen los siguientes valores como los esperados para la configuración óptima:

- Disponibilidad del 98% ± 2%
- Uso de CPU de aproximadamente 90%
- Error en las peticiones menor a 1%

Entorno de prueba:

Entorno	Máquina virtual con 2 vCPU + 2GB Ram + 20 GB Almacenamiento
Sistema operativo	Sistema operativo Ubuntu 22
Base de datos	PostgreSQL en contenedor
Lenguaje API / Framework	Python / Flask
Método de despliegue	Contenedores de Docker
Método de comunicación	HTTP remoto. 80-100ms de latencia esperada
Software de pruebas	JMeter

Tabla 1 - Condiciones del entorno de pruebas

Artefactos de las pruebas
Pool de 4 videos de entre 20mb a 25mb de peso y 4 a 5 minutos de duración
Lista de 100 usuarios registrados en la plataforma
Tablas de datos de videos y tareas vacías

Tabla 2 - Artefactos de prueba

Metodología

Paso	Descripción
1	Se preselecciona un usuario aleatorio del pool de usuarios
2	Autenticar con los datos del usuario para obtener un token de autorización
3	Realizar la petición para cargar un video aleatorio con el token conseguido
4	Validar cada medio segundo el estado de la tarea hasta que sea finalizado
5	Medir el tiempo de ejecución de la prueba

Tabla 3 - Escenario de prueba por cada instancia de JMeter

En la tabla 3 se presenta el escenario de pruebas que se ejecutará por cada instancia de JMeter. A partir de este punto se escalará el número de usuarios concurrentes que deciden hacer la solicitud a la plataforma. **El escalamiento de las pruebas se realizará entre 1 a 10 usuarios concurrentes, replicando cada la prueba 3 veces por cada conjunto de usuarios, para un total de 30 pruebas**. En caso de que no se encuentre una variación significativa, o que la máquina todavía recursos disponibles, **se aumentará el número de instancias concurrentes hasta 20 usuarios.**

Análisis de escenarios

A partir de la metodología descrita, se piensa dar respuesta a los siguientes escenarios:

1. Comportamiento de la plataforma ante la carga en simultaneo de videos
2. Capacidad para la edición de videos en batches

Estos escenarios se pueden determinar a partir del escenario de prueba descrito usando el porcentaje de fallas y el tiempo medio de edición de videos respectivamente. Se espera que para ambos escenarios se genere una gráfica que permita relacionar el throughput del sistema y el tamaño del batch de videos, al igual que encontrar cuál es la capacidad de la capa web para atender a los usuarios

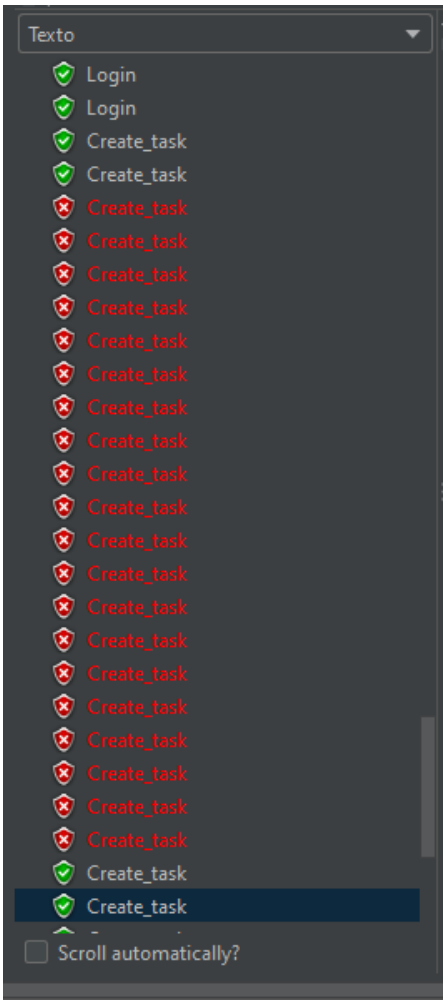
Resultado de prueba:

Al momento de realizar las pruebas, se desarrollaron 4 hilos de ejecución cada uno con 20 usuarios que distribuían las peticiones de login y Create_task (cargar un video) entre 15 y 30 segundos, esto para simular el comportamiento real de distribución de peticiones.

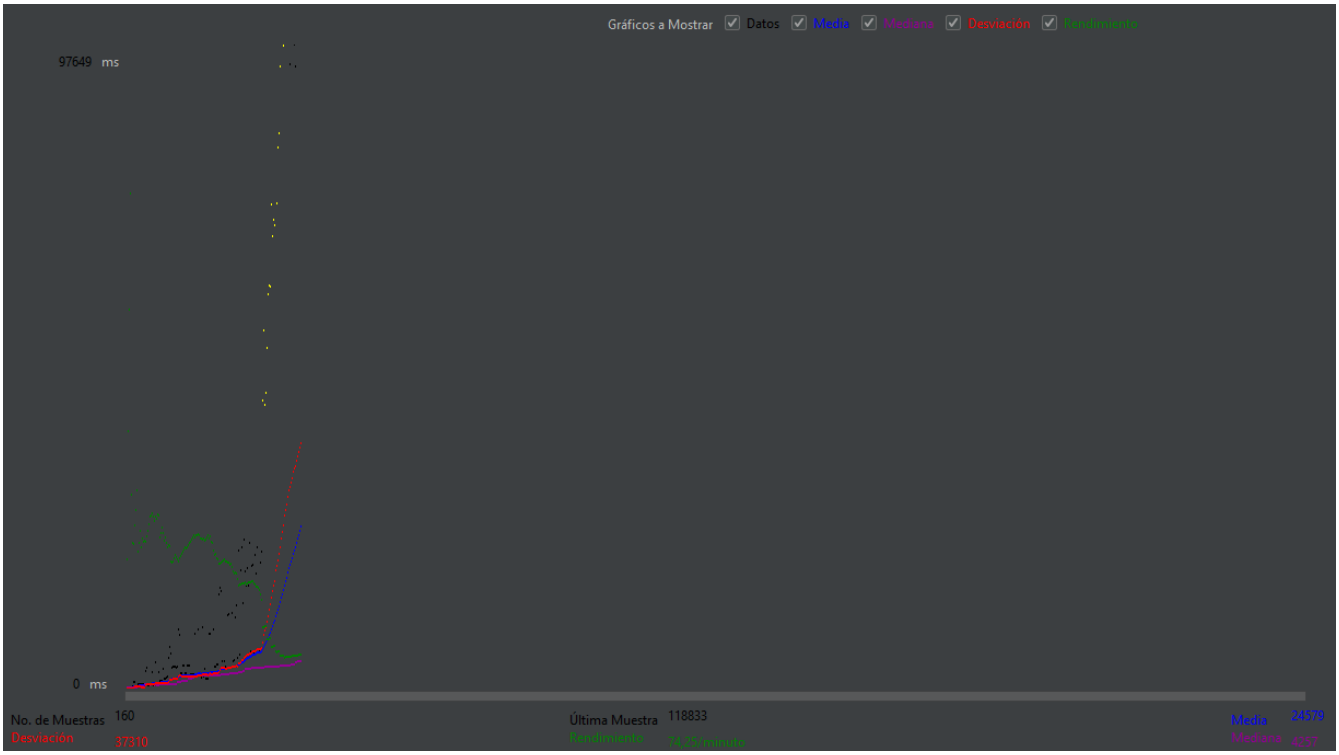
El resumen de resultados de estas pruebas es:

Etiqueta ↓	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec
Login	80	2285	198	6092	1520,21	0,00%	2,3/sec	1,06	0,62
Create_task	80	46873	1134	123848	42281,52	25,00%	37,2/min	0,15	6179,21
Total	160	24579	198	123848	37310,08	12,50%	1,2/sec	0,43	6169,10

Es importante resaltar la resiliencia y recuperación después del fallo que presenta el entorno de trabajo en la nube. Esto debido a que el proceso de create_task por un momento dejo de estar disponible y sin la intervención de terceros este, una vez tuvo disponibilidad para procesar peticiones, continúo recibiendo y respondiendo peticiones. Esto lo podemos evidenciar en la ventana de logs de Jmeter:



Para entender mejor el comportamiento de estas pruebas, se puede acudir a las gráficas generadas por Jmeter. Aquí podemos evidenciar como el proceso inicia con un rendimiento adecuado y a medida que el proceso empieza a cargar las máquinas virtuales con peticiones y videos para procesar, el tiempo de procesamiento aumenta y el rendimiento disminuye drásticamente:



Recomendaciones:

Con los resultados de este experimento, se debe tener en cuenta:

- Al aumentar las peticiones de create_task la cola de actividades aumenta su carga y la del sistema, por ello dividir el sistema y acudir a una arquitectura con microservicios ayudará a controlar mejor los recursos.
- Hay otros recursos en la nube que permiten aumentar la capacidad de los diferentes elementos y servicios de la nube. Acudir a estos ayuda al sistema a mantener la disponibilidad esperada y mejorar la calidad del servicio.
- Actualmente procesar todo el sistema dentro de una máquina virtual puede no ser la mejor solución, hay servicios específicos para ciertas tareas como colas o procesamiento de videos en las diferentes nubes disponibles. Estos servicios podrían ayudar a reducir costos y aumentar la calidad del servicio acelerando los tiempos de desarrollo.